Instructions for Lab #4: Convection

Jonathan Gilligan

2021-02-15

Contents

Exercises with lapse rate, clouds, and water-vapor feedback.

In this lab, we will continue using MODTRAN and will introduce a new climate model, RRTM. RRTM, which stands for "Rapid Radiative Transfer Model," is a radiative-convective model that uses code from the radiative-transfer portion of a state-of-the-art globam climate model called the "Community Climate System Model," developed at the National Center for Atmospheric Research in Boulder CO.

The entire CCSM model runs on giant supercomputers, but this radiative transfer module can run efficiently on an ordinary computer. In order to speed up the calculations, RRTM does not calculate the entire longwave spectrum the way MODTRAN does, but uses a simplified approximation that is much faster when the big climate models need to run this radiative transfer calculation roughly 52 quadrillion (5.2×10^{15}) times in a simulation of 100 years of the earth's climate.

An advantage that RRTM has over MODTRAN, is that MODTRAN assumes that the atmosphere is static (none of the air moves), whereas RRTM allows for convective heat flow. This makes RRTM more realistic, even though it sacrifices detail in its treatment of longwave radiation.

We will use RRTM to explore the role of convection in the earth system and to examine the water-vapor feedback in the presence of convection.

You can run the RRTM model interactively on the web at http://climatemodels.uchicago.edu/rrtm/index.html and I have also written a script that allows you to run it from R.

To run the model interactively, you can adjust various parameters, such as the brightness of the sun, the albedo (it gives you a choice of many natural and human-made surfaces, such as asphalt, concrete, forest, grassland, snow, ocean, and the average for the earth) the concentrations of CO_2 and methane, the relative humidity, and the amount and type of high (cirrus) and low (stratus) clouds.

You can also introduce aerosols typical of different parts of the earth, such as cities (with soot, sulfates, and other pollution), deserts (with blowing dust), oceans (with sea spray and salt), and a Pinatubo-like volcanic eruption.

Like MODTRAN, the model does not automatically adjust the surface temperature. Instead, it calculates the upward and downward flux of longwave and shortwave radiation at 51 different levels of the atmosphere and reports whether the heat flow is balanced (heat in = heat out) at the top of the atmosphere.

If the earth is *gaining* heat, you can manually *raise* the surface temperature until you balance the heat flow, and if the earth is *losing* heat, you can manually *lower* the temperature.

R Interface to RRTM

I have written an R function `run_rrtm` that allows you to manually run RRTM from R. To use this function, you need to include the line `source("_scripts/rrtm.R")` or `source(file.path(script_dir, "rrtm.R"))` to load it.

- `run_rrtm()` allows you to automatically download a file with the data from a MODTRAN run. You call it with the following arguments:
  - `filename` is the name of the file to save the data to. The function returns the output data, so it's optional to specify a filename.
  - `co2_ppm` is the amount of CO_2 in parts per million. The default is 400.
  - `ch4_ppm` is the amount of methane in parts per million. The default is 1.7.
  - `relative_humidity` is the relative humidity, in percent. The default is 80%.
  - `T_surface` is the surface temperature, in Kelvin. The default (for 400 ppm CO_2, etc.) is 284.42. You adjust this to restore radiative equilibrium after you change the parameters (amount of CO_2, lapse rate, etc.).
  - `I_solar` is the brightness of the sun, in Watts per square meter. The default value is 1360.
  - `surface_type` is the type of surface (this is used to calcualte the albedo). The default is 'earth average. The options are:
    - "`earth average`": The average albedo of the earth (0.30)
    - "`asphalt`": Dark asphalt (0.08)
    - "`concrete`": Concrete (0.55)
    - "`desert`": Typical desert (0.40)
    - "`forest`": Typical forest (0.15)
    - "`grass`": Typical grassland (0.25)
    - "`ocean`": Ocean (0.10)
    - "`snow`": Typical snow (0.85)
    - "`ice`": Large ice masses covering ocean or land (0.60)
    - "`soil`": Bare soil (0.17)
    - `"custom": Custom albedo (if you choose this, you need to also supply a value foralbedo'`)
  - `tropopause_km` is the altitude of the tropopause, in kilometers above sea level. The default value is 15. On the earth, the tropopause varies from around 9 km at the poles to around 17 km near the equator.
  - `lapse_rate` is the lapse rate, in Kelvin per kilometer. The default is
    6. The dry adiabatic lapse rate is 10, so it's physically impossible to have a lapse rate greater than 10 and results with `lapse_rate` greater than 10 won't make sense.
  - `low_cloud_frac` is the fraction (from 0–1) of the sky covered by low (stratus) clouds. The default is 0.
  - `high_cloud_frac` is the fraction (from 0–1) of the sky covered by high (cirrus) clouds. The default is 0.
  - `cloud_drop_radius` is the size of the water droplets in the clouds, in microns. The default is 10. (For reference, 10 microns is about the size of a red blood cell). You can reduce this to simulate the indirect aerosol effect.

- `aerosols` allows you to set up the atmosphere with the kinds and quantities of aerosols typical of a number of different environments. Options are:
  - "`none`": No aerosols
  - "`ocean`": Typical ocean aerosols (sea-spray, salt, etc.)
  - "`desert`": Typical desert aerosols (dust, sand)
  - "`city`": Typical city with soot (black carbon) and sulfate aerosols.
  - "`city just sulfates`": Just sulfate aerosols typical of a city.
  - "`city just soot`": Just soot (black carbon) aerosols typical of a city.
  - "`land`": Typical rural land (dust, etc.)
  - "`polluted land`": Typical rural land suffering from pollution (e.g., from farming)
  - "`antarctic`": Typical aerosols for Antarctica
  - "`volcano`": Similar sulfate and dust to the Mt. Pinatubo volcanic eruption.

Any arguments you don't specify explicitly take on their default value. Thus, `run_rrtm(co2_ppm = 800, relative humidity = 10, T_surface = 300)` would run with all the default values, except for 800 ppm CO_2, relative humidity of 10%, and a surface temperature of 300 Kelvin.

`run_rrtm` returns a list of data containing:

- Basic parameters of the model run:
  - `T_surface`
  - `co2_ppm`
  - `ch4_ppm`
  - `I_solar`
  - `albedo`
  - `lapse_rate`
  - `tropopause_km`
  - `relative_humidity`
  - `aerosols`,
  - `low_cloud_frac`
  - `high_cloud_frac`
  - `cloud_drop_radius`
- Results of the model calculations:
  - `Q`: The heat imbalance $I_{\text{in}} - I_{\text{out}}$
  - `i_in`: The net solar radiation absorbed by the earth ($(1-\alpha)I_{\text{solar}}/4$)
  - `i_out`: The net longwave radiation emitted to space from the top of the atmosphere
  - `profile`: A tibble containing a profile of the atmosphere (altitude in km, pressure in millibar, and temperature in Kelvin)
  - `fluxes`: A tibble containing the fluxes (in Watts per square meter) of longwave, shortwave, and total radiation going up and down at 52 levels from the surface to the top of the atmosphere. The columns are `altitude` (km), `T` (temperature in K), `P` (pressure in millibar), `sw_up` (upward shortwave), `sw_down` (downward shortwave), `lw_up` (upward longwave), `lw_down` (downward longwave), `total_up` (`sw_up` + `lw_up`), and `total_down` (sw_down+lw_down').

There are also functions for reading RRTM data files and plotting RRTM data:

- `read_rrtm(file)` reads an RRTM file saved by `run_rrtm` and returns a list of data just like the one returned by `run_rrtm`.
- `plot_heat_flows()`: plots the upward and downward fluxes of radiation from an RRTM file or data structure. You can call it either with `plot_rrtm(file = "filename")` or `plot_rrtm(data = rrtm_data)`, where "`filename`" and "`rrtm_data`" stand for your own filename or rrtm data structure returned by `run_rrtm` or `read_rrtm`.

  You can also specify which wavelengths to plot. By default, it plots shortwave (SW), longwave (LW), and total (SW + LW), but you can specify one or more of `sw = FALSE`, `lw = FALSE`, or `total = FALSE` to omit wavelengths.

Example of running RRTM

Here is an example of running RRTM:

```
default_rrtm = run_rrtm()

# Surface temperature:
default_rrtm$T_surface
```

```
## [1] 284.42
```

```
# Heat imbalance:
default_rrtm$Q
```

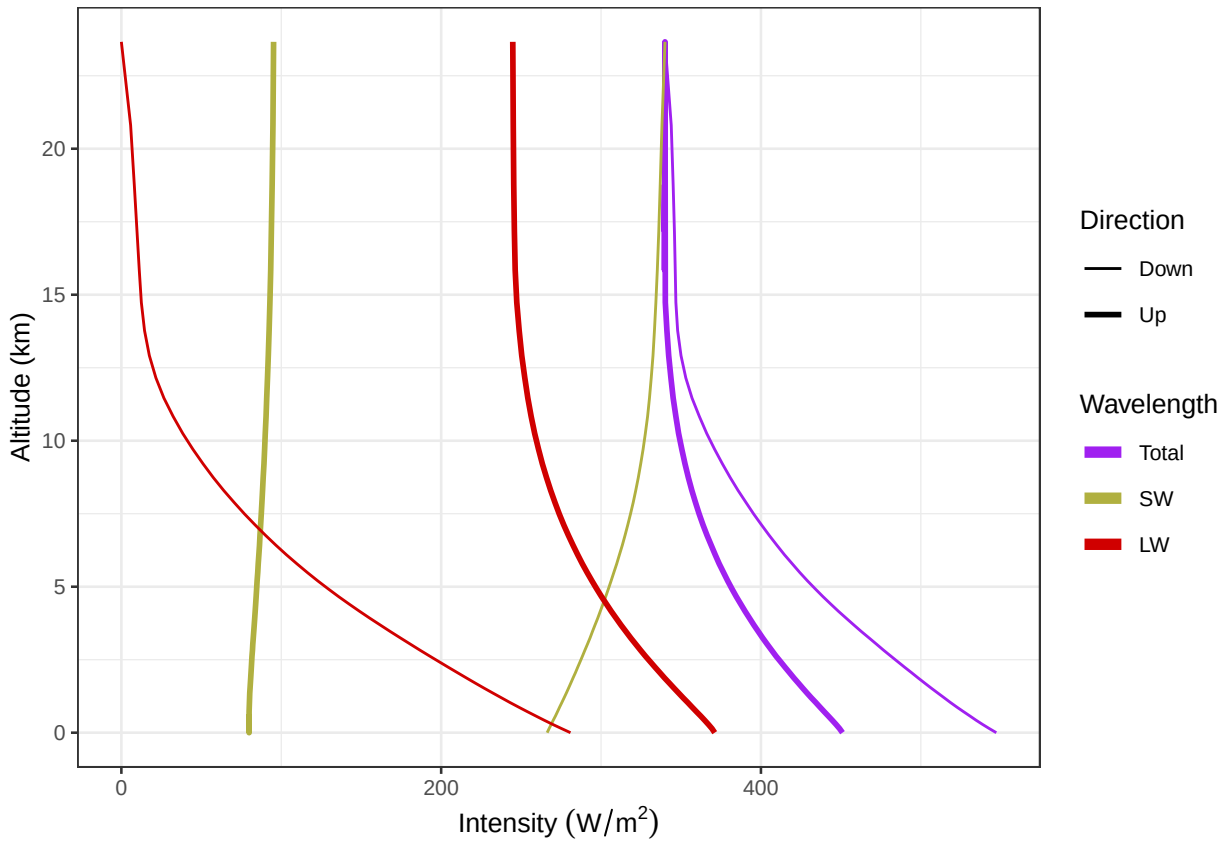
```
## [1] 0
```

This run has surface temperature 284. K and a heat imbalance of 0 Watts per square meter.

Interpreting RRTM Results

We can plot the heat flows as a function of altitude:

```
plot_heat_flows(default_rrtm)
```



What you see in this plot are thick lines representing downward heat flow and thin lines representing upward flow. The different colors represent shortwave, longwave, and total (shortwave + longwave).

A few things to notice: At the top of the atmospere, at 20. km, there is very little longwave going down, but a lot of shortwave going down (around 300. W/m^2). Conversely, there is a modest amount of shortwave going up (around 100. W/m^2), but a lot of longwave going up (around 200. W/m^2).

The upward shortwave radiation is sunlight reflected from the atmosphere and the earth's surface.

The upward longwave radiation is emitted from the surface and the atmosphere. You can see that the longwave radiation, both up and down, is greater closer to the surface, where temperatures are warmer, and smaller at higher altitudes, where the atmosphere is cooler.

If we look at the total radiation, we see that there is a good balance near the top of the atmosphere (the upward and downward lines come together), but in the lower atmosphere, there is a serious imbalance with downward fluxes significantly larger than the upward ones.

This is a consequence of convection: The difference between the downward and upward radiative fluxes is taken up by convection, which moves heat upward when warm air rises and cool air sinks.

Determining Climate Sensitivity with RRTM

We can also use the RRTM model to study what happens when we double CO_2:

```
rrtm_double_co2 = run_rrtm(co2_ppm = 800)
```

When we double CO_2 without changing the surface temperature (T~surface = 284. K), this creates a heat imbalance of 4.20 W/m^2. We can use the online interactive version of RRTM to adjust surface temperature until the heat flows balance. The surface temperature where this happens is 287. K and we can paste it into our R code:

```
new_ts = 286.9 # Kelvin
rrtm_double_co2_balanced = run_rrtm(co2_ppm = 800, T_surface = new_ts)
```

When we set $T_{surface}$ to 287. K, the heat imbalance becomes 0 Watts/m^2. The climate sensitivity is the change in equilibrium $T_{surface}$ when you double CO_2: $\Delta T_{2\times CO_2} = 287.\text{K} - 284.\text{K} = 2.48$ K. You may remember that when we calculated the climate sensitivity with MODTRAN (using constant relative humidity to enable water-vapor feedback) we got $\Delta T_{2\times CO_2} = 1.21$ K for the tropical atmosphere (it's smaller for the other atmospheres), so this shows that including convection in our calculations roughly doubles the climate sensivity.

New R and RMarkdown tricks

Sometimes you may want to use different text into your document, depending on what the result of a calculation is.

For instance, I might have a function called `foo` that returns a number and I want to write something different if `foo(x)` > x than if `foo(x)` < x. Here, the function `ifelse` can come in handy.

```
foo = function(x) {
  x^2
}
```

Now I can write `ifelse(foo(x) < x, "less than", "greater than")`: When x = 0.5, `foo(x)` is less than x, but when x = 2.0, `foo(x)` is greater than x.

You may have spotted a problem with the code above: What if `foo(x) = x`? Then I need another `ifelse`: `ifelse(foo(x) < x, "less than", ifelse(foo(x) > x, "greater than", "equal to"))`. This is cumbersome to type into your text, so you might want to write a function:

```r
compare_f = function(f, x) {
  # f is a function
  # x is a number or a numeric variable
  result = f(x)
  ifelse(result < x, "less than",
         ifelse(result > x, "greater than",
                "equal to"))
}
```

Now I can just write `compare_f(foo, x)`: When x = 0.5, `foo(x)` is less than x, but when x = 2.0, `foo(x)` is greater than x and when x = 1.0, `foo(x)` is equal to x.

This may seem kind of strange, but it can be helpful if you have a report that you prepare regularly with different data and want to be able to update by running RMarkdown with different data sets, and have the text adjust automatically to what the new numbers are.

This approach is used when businesses need to generate monthly reports (e.g., sales, finances, etc.) and want to automate the process with an RMarkdown report template that can be used over and over with each new month's data.

It is also applicable to climate science, where many laboratories like to update their reports every month or every year with the latest climate data.

Exercises

These are the exercises you will work for the lab this week.

General Instructions

In the past three weeks, we focused on mastering many of the basics of using R and RMarkdown. For this week's lab, when you write up the answers, I would like you to think about integrating your R code chunks with your text.

For instance, you can describe what you're going to do to answer the question, and then for each step, after you describe what you're going to do in that step, you can include an R code chunk to do what you just described, and then the subsequent text can either discuss the results of what you just did or describe what the next step of the analysis will do.

This way, your answer can have several small chunks of R code that build on each other and follow the flow of your text.

Chapter 5 Exercise

For this model, you will use the RRTM model, which includes both radiation and convection.

Exercise 5.1: Lapse Rate

Run the RRTM model in its default configuration and then vary the lapse rate from 0 to 10 K/km. For each value of the lapse rate, adjust the surface temperature until the earth loses as much heat as it gains (i.e., the value of Q in the `run_rrtm` model output is zero.)

It will probably be easier to do this with the interactive version of the RRTM model at http://climatemodels.uchicago.edu/rrtm/ than with the R interface `run_rrtm`.

a) Make a tibble containing the values of the lapse rate and the corresponding equilibrium surface temperature, and make a plot with lapse rate on the horizontal axis and surface temperature on the vertical axis.

b) Describe how the equilibrium surface temperature varies as the lapse rate varies.